

To: Distribution

Date: March 31, 1983

From: Harry B. Stewart

Subject: Speech Handler Interface

Here is a preliminary specification for a parallel bus speech handler to support the SC01 speech chip in future products. Comments are invited.

cc: Ken Balthaser
Sherwin Gooch
Rashid Khan
Joe Miller
Rick Nordin
Rich Pasco
Scott Scheiman

SPEECH HANDLER PRELIMINARY SPECIFICATION

Harry B. Stewart
March 31, 1983

TABLE OF CONTENTS

Introduction

Hardware assumptions

Handler functionality

Handler/CIO interface

Speech data formats

Implementation details

Appendix A -- World English Spelling Form

Appendix B -- VOTRAX SC01 Symbolic Form

Appendix C -- Translate Table Format

Appendix D -- Translate Table for SC01 Symbolic Form

Appendix E -- Translate Tabale for WES Form

SPEECH HANDLER PRELIMINARY SPECIFICATION

Harry B. Stewart
March 31, 1983

Introduction

There are several levels of speech representation which might be used to drive an SC01 speech device, four of which are shown below:

'hello' -- Text representation of "hello".

'heloe' -- World English Spelling of "hello".

'H EH1 EH2 L O1 PA0' -- Votrax symbolic phoneme representation of "hello".

1B 02 01 18 35 03 -- Votrax numeric phoneme representation of "hello", in hexadecimal.

It is my recommendation that we support the lower three levels shown, mapped to the SC01 speech chip, the device used in the computer products currently being developed.

In order to allow for upward mobility of application and user software to other speech chips which might be utilized in Atari's future products, all speech data will be identified as being of a specific (device dependent) type and form. Future products will be able to accept data of their native type as well as translate data of older types to the closest equivalent sound in their new hardware.

Thus, for example, the three forms supported for the SC01 might be called '1P' (type #1, phonetic form), '1S' (type #1, symbolic form) and '1N' (type #1, numeric form), and future type/forms for the SC02 (or any other device) might be called '2P', '2S' and '2N'. Perhaps even a '2T' (type #2, text form) might be supported.

Hardware assumptions

The list items represent assumptions about the hardware interface, NOT desires.

A VOTRAX SC01 speech chip will be utilized.

One phoneme (6 bits) of external buffering is provided.

The computer has direct control of the SC01 STB line.

A status bit and disableable IRQ interrupt is associated with the SC01 A/R line.

There is no pitch/inflection control.

There is no volume control.

Handler functionality

The handler shall be able to output speech data (phonemes) when presented with data in any of three user formats: phonetic, symbolic or numeric. In the phonetic and symbolic formats, a phoneme is represented by one to three ATASCII characters followed by a delimiter character; the handler converts each symbolic phoneme to one or more six bit numeric phoneme codes. In the numeric format, a phoneme is comprised of the lower six bits of an eight bit byte, which is assumed to contain the numeric phoneme code.

In addition, the handler shall have three output modes: direct, semi-buffered and fully-buffered. The output characteristics of the three modes are described below.

Direct -- output phoneme, wait for done, then return to caller.

Semi-buffered -- wait for prior phoneme done, output new phoneme, then return to caller.

Fully-buffered -- return to caller, then output all phonemes at interrupt level.

The caller may determine the completion of a phoneme string or synchronize himself to specific phonemes by any of several techniques, as listed below.

Phoneme counter in database -- The handler shall maintain a one byte counter in the PBI database area which shall be incremented as each phoneme is strobed into the SC01. This variable may be examined and/or altered by the caller at will.

FIFO control in database -- There shall be a FIFO associated with the handler which shall be used when phoneme output is to be fully buffered. This FIFO and its control elements are resident in the PBI database area and may be examined (but not altered) by the caller at will.

Markers -- There shall be one or more codes reserved for "markers". When a marker is about to be processed as phoneme data, the handler shall increment a marker variable in the PBI database area and produce a null phoneme (zero

duration).

Which technique to use is a function of: 1) the output mode selected, 2) how well the caller records map to the synchronization points, and 3) whether the handler is called directly by the user or is invoked through CIO.

Handler/CIO interface

The device name is 'V:'

OPEN

The IOCB buffer pointer shall point to a sequence of ATASCII characters of the form shown.

V:<type><form><mode><EOL>

where: <type> = 'l'
 <form> = 'P' for World English Spelling phonetic form
 'S' for Votrax symbolic form,
 'N' for Votrax numeric form,
 'U' for user specified symbolic form.
 <mode> = 'D' for direct mode,
 'S' for semi-buffered mode,
 'F' for fully buffered mode.

In response to an OPEN command, the handler initializes the output FIFO, resets all handler database variables and flags, initializes the translate table pointer, and sends a STOP phoneme to the SC01. If the SC01 does not respond with an A/R status within xx milliseconds, an error status is returned.

If the form is 'U', the user is responsible for providing the address of his translate table by storing it directly into the handler's data base or by issuing a SPECIAL command after the OPEN.

If not specified, the type, form and mode default as shown below.

<type> = 'l'.
<form> = 'N'.
<mode> = 'D'.

Error conditions:

xx -- Votrax not responding.
xx -- Invalid type (not equal to 'l').
xx -- Invalid form (not equal to 'P', 'N', 'S' or 'U').
xx -- Invalid mode (not equal to 'D', 'S' or 'F').

CLOSE

In response to a CLOSE command, the handler sends a STOP phoneme to the SC01.

Error conditions:

none

PUT BYTE

The handler accepts a single byte (character) of phonetic form, symbolic form, user form or numeric form data. The data is in the A register; the handler returns status in the Y register. The table below shows the processing that occurs for each byte, depending upon the form and mode.

FORM	MODE	PROCESS
N	D	The phoneme is output to the SC01, and the handler waits for the next SC01 request before returning.
	S	The handler waits for the SC01 to request a byte, outputs the phoneme to the SC01 and then returns.
	F	The byte goes to the output FIFO. If the FIFO is full, the handler loops waiting for an empty slot.
P,S,U	D	A token is assembled, translated to the proper numeric code, and then proceeds as in N-D above.
	S	a token is assembled, translated to the proper numeric code, and then proceeds as in N-S above.
	F	a token is assembled, translated to the proper numeric code, and then that phoneme is inserted to the output FIFO.

When operating in the fully buffered mode, the handler enables the SC01 interrupt, so that the handler's interrupt service routine can empty the FIFO.

Error conditions:

xx -- Unrecognizable text token.

SPECIAL

Define new FIFO (need location and size) -- TBD.

Define new translate table (need location) -- TBD.

INIT

TBD.

Speech data formats

World English Spelling -- "hello" is shown encoded in WES format, which requires 6 characters (see Appendix A for more information).

```
+--+--+--+--+
|H E L O E |
+--+--+--+--+
```

For the P form, the following characters are treated as word delimiters and produce pauses:

- ' ' -- space produces a short pause.
- ',' -- comma produces an short pause (intermediate when followed by a space).
- ',' -- period produces a long pause.
- '?' -- question mark produces a long pause.

For the P form, the hyphen is a token delimiter which allows the handler to unambiguously process the multi-character tokens. For example the word 'mishap' would be spelled 'mis-hap'.

For the P form, the ATASCII EOL is ignored.

The asterisk character is treated as a marker by the handler.

Symbolic -- "hello" is shown encoded in Votrax symbolic format, which requires 18 characters (see Appendix B for more information).

```
+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+
|H  E H 1  E H 2  L  O 1  P A Ø|
+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+
```

For the S form, the following characters are treated as token delimiters and do not produce or alter phonemes:

```
' ' -- space
',' -- comma.
'.' -- period.
'? ' -- question mark.
'- ' -- hyphen.
<EOL> -- ATASCII end of line.
```

The asterisk character is treated as a marker by the handler.

Numeric -- "hello" is shown encoded in Votrax numeric format, which requires 6 bytes.

```
+---+---+---+---+---+---+
|1B 02 01 18 35 03|
+---+---+---+---+---+---+
```

All bytes received by the handler are truncated to 6 bit phoneme values and sent to the SC01, with the following two exceptions:

```
$9B is an EOL and is ignored by the handler (null operation).
$7F is the code for a marker.
```

Implementation details

FIFO related items:

If the FIFO fills in mid-record, the handler waits until there is room in the FIFO for the next phoneme.

FIFO size is limited to 255 items (phonemes and markers). The default FIFO contains up to 16 items.

Translate table related items (see Appendix C for more information):

Each translate table is limited to 256 bytes.

RAM memory utilization (32 bytes available):

```
Phoneme counter [1].
Marker flag [1]

FIFO input index [1].
FIFO output index [1].
FIFO size [1].
FIFO base pointer [2].
```


Default FIFO buffer [16].

Current data type [1].

Current data form [1].

Current output mode [1].

Translate table base pointer [2].

Translate table offset [1].

Temporaries [3].

Appendix A -- World English Spelling Form

speech token	sound	Votrax equivalent
a	fAt	AE /2E or AE1/2F
aa	fAther	AH1/15
ae	pAY	A2 /05, A1 /06, A /20 or AY /21
ar	fAR	??
au	tAUt	AW1/13, AW2/30 or AW /3D
b	But	B /0E
ch	CHum	T /2A + CH /10
d	Dig	D /1E
e	sEt	EH3/00, EH2/01, EH1/02 or EH /3B
ee	sEE	Y /29, or E /2C or E1 /3C
er	gathER	ER /3A
f	Fat	F /1D
g	Gum	G /1C
h	Hat	H /1B
i	In	I3 /09, I2 /0A, I1 /0B or I /27
ie	tIE	??
j	Jam	D /1E + J /1A
k	Kit	K /19
l	Let	L /18
m	Met	M /0C
n	Net	N /0D
ng	siNG	NG /14
nk	siNK	??
o	On	??
oe	tOE	O /26, O2 /34 or O1 /35
oi	OIl	??
oo	tOO	U /28, IU /36 or U1 /37
or	fOR	??
ou	OUt	??
p	Pet	P /25
r	Run	R /2B
s	Set	S /1F
sh	SHed	SH /11
t	Tin	DT /04 or T /2A
th	THis	THV/38
thh	THing	TH /39
u	Up	UH1/32 or UH /33
ue	hUE	??
ur	fUR	ER /3A
uu	bOOk	OO1/16 or OO /17
v	Van	V /0F
w	Win	W /2D
wh	WHen	??
y	Yes	Y1 /22
z	Zoo	Z /12
zh	viSion	ZH /07

Appendix B -- Votrax SC01 Symbolic Form

speech token	sound	Votrax numeric equivalent
EH3	jackEt	00
EH2	Enlist	01
EH1	hEAvy	02
PA0	<pause>	03
DT	buTTer	04
A2	mAde	05
A1	mAde	06
ZH	aZure	07
AH2	hOnest	08
I3	inhibIt	09
I2	Inhibit	0A
I1	inhIbit	0B
M	Mat	0C
N	suN	0D
B	Bag	0E
V	Van	0F
CH	CHip	10
SH	SHop	11
Z	Z00	12
AW1	lAWful	13
NG	thiNG	14
AH1	fAther	15
OO1	lOOking	16
OO	bOOk	17
L	Land	18
K	triCK	19
J	JuDGe	1A
H	Hello	1B
G	Get	1C
F	Fast	1D
D	paID	1E
S	paSS	1F
A	dAY	20
AY	dAY	21
Y1	Yard	22
UH3	missIOn	23
AH	mOp	24
P	Past	25
O	cOld	26
I	pIn	27
U	mOve	28
Y	anY	29
T	Tap	2A
R	Red	2B
E	mEEt	2C
W	Win	2D
AE	dAd	2E

AE1	After	2F
AW2	sAlty	30
UH2	About	31
UH1	Uncle	32
UH	cUp	33
O2	fOr	34
O1	abOArd	35
IU	yOU	36
U1	yOU	37
THV	THe	38
TH	THin	39
ER	bIRd	3A
EH	gEt	3B
E1	bE	3C
AW	cAll	3D
PA1	<pause>	3E
STOP	<stop>	3F

Appendix C -- Translate Table Format

The speech translate table is a state table that allows the translator to be implemented as a finite state machine (FSM). The advantages of this approach are twofold: 1) the table is very compact, and 2) the FSM requires character storage for only one character at a time, rather than the characters for one complete token at a time.

The translate table consists of a collection of multiple byte entries. Each entry consists of a match character followed by a match directive which is to be executed if the input character matches the match character. In general, the match directive will advance the FSM to a new state, and may in addition produce one or more output phonemes.

The formats for the match byte and directive byte are shown below.

Match byte:	+--+--+--+--+--+--+--+	
	0 match char	match character
	+--+--+--+--+--+--+--+	
	+--+--+--+--+--+--+--+	
	1 xxxx	NIL
	+--+--+--+--+--+--+--+	
Directive byte:	^+--+--+--+--+--+--+--+	
	0 0 phoneme	SC01 phoneme code
	+--+--+--+--+--+--+--+	
	+--+--+--+--+--+--+--+	
	0 1 0 n	n phonemes follow
	+--+--+--+--+--+--+--+	
	+--+--+--+--+--+--+--+	
	0 1 1 code	special action code
	+--+--+--+--+--+--+--+	(see Note 1, below)
	+--+--+--+--+--+--+--+	
	1 offset	offset to next state
	+--+--+--+--+--+--+--+	(see Note 2, below)

Note 1 -- Special actions include the following:

- delimiter (ignore).
- generate a marker.
- error (invalid token).

Note 2 -- The offset is used to update the translate table index as follows:

$\text{index} := (\text{index} + \text{offset}) \bmod 256$

This implies the following:

The maximum table size is 256 bytes.

All state transitions are in the forward direction, except for the transition to the top level state which is implicit in the specification of the FSM.

A state transition destination must be within 127 bytes of the source directive.

The FSM scanner operates with the input data as described below.

1. The translate table index is set to the beginning of the translate table (=0).
2. A new character is obtained.
3. The scanner scans the table linearly trying to find a match between the token character and one of the table entry match characters before a NIL entry is seen.
4. If a match is found, the scanner processes the match directive and does one of the following actions, based upon the type of directive.
 - a. If the directive is a phoneme or multiple phonemes, the phoneme(s) are output and scanning proceeds at step 1.
 - b. If the directive is a special action, the specified action is taken and scanning proceeds at step 1.
 - c. If the directive is a table offset (new state), the table index is updated as specified and scanning proceeds at step 2.
5. If a match is not found (NIL found first), the scanner processes the match directive associated with the NIL and does one of the following actions, based upon the type of directive.
 - a. If the directive is a phoneme or multiple phonemes, the phoneme(s) are output and scanning proceeds at step 6.
 - b. If the directive is a special action, the specified action is taken and scanning proceeds at step 6.
 - c. If the directive is a table offset (new state), the table

index is updated as specified and scanning proceeds at step 2.

6. The table pointer is set to the beginning of the translate table (this is the same as step 1).

7. Scanning proceeds at step 3, using the character that produced the NIL match.

The translate table for the SC01 Symbolic Format is shown in Appendix D and the translate table for the World English Spelling Format is shown in Appendix E.

Appendix D -- Translate Table for SC01 Symbolic Form

LABEL (state)	MATCH CHAR	NEXT STATE	DIRECTIVE:	
			SPECIAL ACTION	PHONEME CODE
Start	'A'	Ax		
	'B'			B
	'C'	Cx		
	'D'	Dx		
	'E'	Ex		
	'F'			F
	'G'			G
	'H'			H
	'I'	Ix		
	'J'			J
	'K'			K
	'L'			L
	'M'			M
	'N'	Nx		
	'O'	Ox		
	'P'	Px		
	'R'			R
	'S'	Sx		
	'T'	Tx		
	'U'	Ux		
	'V'			V
	'W'			W
	'Y'	Yx		
	'Z'	Zx		
	' '		<delim>	
	'.'		<delim>	
	'.'		<delim>	
	'?'		<delim>	
	'-'		<delim>	
	'*'		<marker>	
	NIL		<error>	
Ax	'E'	AEx		
	'H'	AHx		
	'W'	AWx		
	'Y'			AY
	'1'			A1
Cx	'2'			A2
	NIL			A
	'H'			CH
Dx	NIL		<error>	
Ex	'T'			DT
	NIL			D
Ex	'H'	EHx		
	'R'			ER
	'1'			E1
	NIL			E

Ix	'U'		IU
	'1'		I1
	'2'		I2
	'3'		I3
	NIL		I
Nx	'G'		NG
	NIL		N
Ox	'O'	OOx	O1
	'1'		O2
	'2'		O
	NIL		
Px	'A'	PAx	P
	NIL		SH
Sx	'H'		
	'T'	STx	S
	NIL		
Tx	'H'	THx	T
	NIL		
Ux	'H'	UHx	U1
	'1'		U
	NIL		Y1
Yx	'1'		Y
	NIL		ZH
Zx	'H'		Z
	NIL		AE1
AEx	'1'		AE2
	'2'		AE
	NIL		AH1
AHx	'1'		AH2
	'2'		AH
	NIL		AW1
AWx	'1'		AW2
	'2'		AW
	NIL		EH1
EHx	'1'		EH2
	'2'		EH3
	'3'		EH
	NIL		OO1
OOx	'1'		OO
	NIL		PA0
PAx	'0'		PA1
	'1'		
	NIL		
STx	'O'	STOx	<error>
	NIL		<error>
THx	'V'		THV
	NIL		TH
UHx	'1'		UH1
	'2'		UH2
	'3'		UH3
	NIL		UH
STOx	'P'		STOP
	NIL		
			<error>

Appendix E -- Translate Table for World English Spelling Form

The WES phoneme codes marked with an asterisk have yet to be mapped to their SC01 equivalents; some of these will be single SC01 codes and some will require multiple SC01 codes.

LABEL (state)	MATCH CHAR	NEXT STATE	DIRECTIVE:	
			SPECIAL ACTION	PHONEME CODE
Start	'A'	Ax		
	'B'			B
	'C'	Cx		
	'D'			D
	'E'	Dx		
	'F'			F
	'G'			G
	'H'			H
	'I'	Ix		
	'J'			J
	'K'			K
	'L'			L
	'M'			M
	'N'	Nx		
	'O'	Ox		
	'P'			P
	'R'			R
	'S'	Sx		
	'T'	Tx		
	'U'	Ux		
	'V'			V
	'W'	Wx		
	'Y'			Yl
	'Z'	Zx		
	' '			PA0
	'.'			PA0
	'.'			PA1
	'?'			PA1
	'-'		<delim>	
	'*'		<marker>	
	NIL		<error>	
Ax	'A'			AHl
	'E'			ae *
	'R'			ar *
	'U'			au *
Cx	NIL			a *
	'H'			ch *
Ex	NIL		<error>	
	'E'			ee *
	'R'			ER
	NIL			e *

Ix	'E'		ie *
	NIL		i *
Nx	'G'		NG
	'K'		nk *
	NIL		N
Ox	'E'		oe *
	'I'		oi *
	'O'		oo *
	'R'		or *
	'U'		ou *
	NIL		o *
Sx	'H'		SH
	NIL		S
Tx	'H'	THx	
	NIL		T
Ux	'E'		ue *
	'R'		ER
	'U'		uu *
	NIL		u *
Wx	'H'		wh *
	NIL		W
Zx	'H'		ZH
	NIL		Z
THx	'H'		THV
	NIL		TH

To: Distribution

Date: March 31, 1983

From: Harry B. Stewart

Subject: Speech Handler Interface

Here is a preliminary specification for a parallel bus speech handler to support the SC01 speech chip in future products. Comments are invited.

cc: Ken Balthaser
Sherwin Gooch
Rashid Khan
Joe Miller
Rick Nordin
Rich Pasco
Scott Scheiman

SPEECH HANDLER PRELIMINARY SPECIFICATION

Harry B. Stewart
March 31, 1983

TABLE OF CONTENTS

Introduction

Hardware assumptions

Handler functionality

Handler/CIO interface

Speech data formats

Implementation details

Appendix A -- World English Spelling Form

Appendix B -- VOTRAX SC01 Symbolic Form

Appendix C -- Translate Table Format

Appendix D -- Translate Table for SC01 Symbolic Form

Appendix E -- Translate Tabale for WES Form

SPEECH HANDLER PRELIMINARY SPECIFICATION

Harry B. Stewart
March 31, 1983

Introduction

There are several levels of speech representation which might be used to drive an SC01 speech device, four of which are shown below:

'hello' -- Text representation of "hello".

'heloe' -- World English Spelling of "hello".

'H EH1 EH2 L O1 PA0' -- Votrax symbolic phoneme representation of "hello".

1B 02 01 18 35 03 -- Votrax numeric phoneme representation of "hello", in hexadecimal.

It is my recommendation that we support the lower three levels shown, mapped to the SC01 speech chip, the device used in the computer products currently being developed.

In order to allow for upward mobility of application and user software to other speech chips which might be utilized in Atari's future products, all speech data will be identified as being of a specific (device dependent) type and form. Future products will be able to accept data of their native type as well as translate data of older types to the closest equivalent sound in their new hardware.

Thus, for example, the three forms supported for the SC01 might be called '1P' (type #1, phonetic form), '1S' (type #1, symbolic form) and '1N' (type #1, numeric form), and future type/forms for the SC02 (or any other device) might be called '2P', '2S' and '2N'. Perhaps even a '2T' (type #2, text form) might be supported.

Hardware assumptions

The list items represent assumptions about the hardware interface, NOT desires.

A VOTRAX SC01 speech chip will be utilized.

One phoneme (6 bits) of external buffering is provided.

The computer has direct control of the SC01 STB line.

A status bit and disableable IRQ interrupt is associated with the SC01 A/R line.

There is no pitch/inflection control.

There is no volume control.

THE WAY WE'VE RESERVED RAM FOR PBI SENSORS MAKES ACCESSING THE DATABASE AREA DIFFICULT FOR AN APPLICATION. EACH SLOT NUMBER GETS 1/2 OF THE AREA D600-D6FF. E.G. SLOT 1 GETS D620-D63F. IT'S HARD TO KNOW

Handler functionality

The handler shall be able to output speech data (phonemes) when presented with data in any of three user formats: phonetic, symbolic or numeric. In the phonetic and symbolic formats, a phoneme is represented by one to three ATASCII characters followed by a delimiter character; the handler converts each symbolic phoneme to one or more six bit numeric phoneme codes. In the numeric format, a phoneme is comprised of the lower six bits of an eight bit byte, which is assumed to contain the numeric phoneme code.

In addition, the handler shall have three output modes: direct, semi-buffered and fully-buffered. The output characteristics of the three modes are described below.

Direct -- output phoneme, wait for done, then return to caller.

Semi-buffered -- wait for prior phoneme done, output new phoneme, then return to caller.

Fully-buffered -- return to caller, then output all phonemes at interrupt level.

The caller may determine the completion of a phoneme string or synchronize himself to specific phonemes by any of several techniques, as listed below.

Phoneme counter in database -- The handler shall maintain a one byte counter in the PBI database area which shall be incremented as each phoneme is strobed into the SC01. This variable may be examined and/or altered by the caller at will.

FIFO control in database -- There shall be a FIFO associated with the handler which shall be used when phoneme output is to be fully buffered. This FIFO and its control elements are resident in the PBI database area and may be examined (but not altered) by the caller at will.

Markers -- There shall be one or more codes reserved for "markers". When a marker is about to be processed as phoneme data, the handler shall increment a marker variable in the PBI database area and produce a null phoneme (zero

WHICH SLOT AN ARBITRARY DEVICE IS IN. HOWEVER, I GUESS WE DID ASSIGN SLOTS TO HUBS & SLOTS 1 TO VOICE. WE'VE JUST HAVE TO BE FIRM ON THAT.

duration).

Which technique to use is a function of: 1) the output mode selected, 2) how well the caller records map to the synchronization points, and 3) whether the handler is called directly by the user or is invoked through CIO.

Handler/CIO interface

The device name is 'V:'

OPEN

The IOCB buffer pointer shall point to a sequence of ATASCII characters of the form shown.

V:<type><form><mode><EOL>

where: <type> = 'l'
 <form> = 'P' for World English Spelling phonetic form,
 'S' for Votrax symbolic form,
 'N' for Votrax numeric form,
 'U' for user specified symbolic form.
 <mode> = 'D' for direct mode,
 'S' for semi-buffered mode,
 'F' for fully buffered mode.

In response to an OPEN command, the handler initializes the output FIFO, resets all handler database variables and flags, initializes the translate table pointer, and sends a STOP phoneme to the SC01. If the SC01 does not respond with an A/R status within xx milliseconds, an error status is returned.

If the form is 'U', the user is responsible for providing the address of his translate table by storing it directly into the handler's data base or by issuing a SPECIAL command after the OPEN.

If not specified, the type, form and mode default as shown below.

<type> = 'l'.
<form> = 'N'.
<mode> = 'D'.

Error conditions:

xx -- Votrax not responding.
xx -- Invalid type (not equal to 'l').
xx -- Invalid form (not equal to 'P', 'N', 'S' or 'U').
xx -- Invalid mode (not equal to 'D', 'S' or 'F').

CLOSE

In response to a CLOSE command, the handler sends a STOP phoneme to the SC01.

Error conditions:

none

PUT BYTE

The handler accepts a single byte (character) of phonetic form, symbolic form, user form or numeric form data. The data is in the A register; the handler returns status in the Y register. The table below shows the processing that occurs for each byte, depending upon the form and mode.

FORM	MODE	PROCESS
N	D	The phoneme is output to the SC01, and the handler waits for the next SC01 request before returning.
	S	The handler waits for the SC01 to request a byte, outputs the phoneme to the SC01 and then returns.
	F	The byte goes to the output FIFO. If the FIFO is full, the handler loops waiting for an empty slot.
P,S,U	D	A token is assembled, translated to the proper numeric code, and then proceeds as in N-D above.
	S	a token is assembled, translated to the proper numeric code, and then proceeds as in N-S above.
	F	a token is assembled, translated to the proper numeric code, and then that phoneme is inserted to the output FIFO. AS IN 11-1 ABOVE

When operating in the fully buffered mode, the handler enables the SC01 interrupt, so that the handler's interrupt service routine can empty the FIFO.

Error conditions:

xx -- Unrecognizable text token.

SPECIAL

Define new FIFO (need location and size) -- TBD.

Define new translate table (need location) -- TBD.

INIT

TBD.

Speech data formats

World English Spelling -- "hello" is shown encoded in WES format, which requires 6 characters (see Appendix A for more information).

```
+--+--+--+--+--+
|H E L O E |
+--+--+--+--+--+
```

For the P form, the following characters are treated as word delimiters and produce pauses:

- ' ' -- space produces a short pause.
- ',' -- comma produces an short pause (intermediate when followed by a space).
- ',' -- period produces a long pause.
- '?' -- question mark produces a long pause.

For the P form, the hyphen is a token delimiter which allows the handler to unambiguously process the multi-character tokens. For example the word 'mishap' would be spelled 'mis-hap'.

For the P form, the ATASCII EOL is ignored.

The asterisk character is treated as a marker by the handler.

Symbolic -- "hello" is shown encoded in Votrax symbolic format, which requires 18 characters (see Appendix B for more information).

```
+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+
|H  E H 1  E H 2  L  O 1  P A 0|
+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+
```

For the S form, the following characters are treated as token delimiters and do not produce or alter phonemes:

' ' -- space
 ', ' -- comma.
 '.' -- period.
 '?' -- question mark.
 '-' -- hyphen.
 <EOL> -- ATASCII end of line.

The asterisk character is treated as a marker by the handler.

Numeric -- "hello" is shown encoded in Votrax numeric format, which requires 6 bytes.

```
+--+--+--+--+--+--+--+
|1B 02 01 18 35 03|
+--+--+--+--+--+--+--+
```

All bytes received by the handler are truncated to 6 bit phoneme values and sent to the SC01, with the following two exceptions:

\$9B is an EOL and is ignored by the handler (null operation).
 \$7F is the code for a marker.

Implementation details

FIFO related items:

If the FIFO fills in mid-record, the handler waits until there is room in the FIFO for the next phoneme.

FIFO size is limited to 255 items (phonemes and markers). The default FIFO contains up to 16 items.

*PCI DATABASE RAM LIMITED
TO 32 BYTES PERHAPS THE*

Translate table related items (see Appendix C for more information):

APPL. ATTACH

Each translate table is limited to 256 bytes.

*CONV. IT
WAS, SINCE
(S. 1980)*

RAM memory utilization (32 bytes available):

Phoneme counter [1].
 Marker flag [1]

*OH. YOU ARE AWARE.
NEVER MIND.*

FIFO input index [1].
 FIFO output index [1].
 FIFO size [1].
 FIFO base pointer [2].

Default FIFO buffer [16].

Current data type [1].

Current data form [1].

Current output mode [1].

Translate table base pointer [2].

Translate table offset [1].

Temporaries [3].

Appendix A -- World English Spelling Form

speech token	sound	Votrax equivalent
a	fAt	AE /2E or AE1/2F
aa	fAther	AH1/15
ae	pAY	A2 /05, A1 /06, A /20 or AY /21
ar	fAR	??
au	tAUt	AW1/13, AW2/30 or AW /3D
b	But	B /0E
ch	CHum	T /2A + CH /10
d	Dig	D /1E
e	sEt	EH3/00, EH2/01, EH1/02 or EH /3B
ee	sEE	Y /29, or E /2C or E1 /3C
er	gathER	ER /3A
f	Fat	F /1D
g	Gum	G /1C
h	Hat	H /1B
i	In	I3 /09, I2 /0A, I1 /0B or I /27
ie	tIE	??
j	Jam	D /1E + J /1A
k	Kit	K /19
l	Let	L /18
m	Met	M /0C
n	Net	N /0D
ng	siNG	NG /14
nk	siNK	??
o	On	??
oe	tOE	O /26, O2 /34 or O1 /35
oi	OIl	??
oo	tOO	U /28, IU /36 or U1 /37
or	fOR	??
ou	OUt	??
p	Pet	P /25
r	Run	R /2B
s	Set	S /1F
sh	SHed	SH /11
t	Tin	DT /04 or T /2A
th	THis	THV/38
thh	THing	TH /39
u	Up	UH1/32 or UH /33
ue	hUE	??
ur	fUR	ER /3A
uu	boOk	OO1/16 or OO /17
v	Van	V /0F
w	Win	W /2D
wh	WHen	??
y	Yes	Y1 /22
z	Zoo	Z /12
zh	viSion	ZH /07

'A' = 'a'
CND 11/10/1971

OR WOULD THE
OVERLAP THE
NEW WAVE 1971

MIGHT BE MORE TO
LEFT OF-3 NO WFO-
SOUNDS 5 11

ALSO
- PDS
- DABLE
etc.

DEEN, 11/10/1971
11/10/1971
11/10/1971

Appendix B -- Votrax SC01 Symbolic Form

speech token	sound	Votrax numeric equivalent
EH3	jackEt	00
EH2	Enlist	01
EH1	hEAvy	02
PA0	<pause>	03
DT	buTTer	04
A2	mAdE	05
A1	mAdE	06
ZH	aZure	07
AH2	hOnest	08
I3	inhibIt	09
I2	Inhibit	0A
I1	inhIbit	0B
M	Mat	0C
N	suN	0D
B	Bag	0E
V	Van	0F
CH	CHip	10
SH	SHop	11
Z	Z00	12
AW1	lAWful	13
NG	thiNG	14
AH1	fAther	15
OO1	looKing	16
OO	boOk	17
L	Land	18
K	triCK	19
J	JuDGe	1A
H	Hello	1B
G	Get	1C
F	Fast	1D
D	paID	1E
S	paSS	1F
A	dAY	20
AY	dAY	21
Y1	Yard	22
UH3	missIOOn	23
AH	mOp	24
P	Past	25
O	cOld	26
I	pIn	27
U	mOve	28
Y	anY	29
T	Tap	2A
R	Red	2B
E	mhEt	2C
W	Win	2D
AE	dAd	2E

AE1	After	2F
AW2	sAlty	30
UH2	About	31
UH1	Uncle	32
UH	cUp	33
O2	fOr	34
O1	abOArD	35
IU	yOU	36
U1	yOU	37
THV	THe	38
TH	THin	39
ER	bIRd	3A
EH	gEt	3B
E1	bE	3C
AW	cAll	3D
PA1	<pause>	3E
STOP	<stop>	3F

Appendix C -- Translate Table Format

The speech translate table is a state table that allows the translator to be implemented as a finite state machine (FSM). The advantages of this approach are twofold: 1) the table is very compact, and 2) the FSM requires character storage for only one character at a time, rather than the characters for one complete token at a time.

The translate table consists of a collection of multiple byte entries. Each entry consists of a match character followed by a match directive which is to be executed if the input character matches the match character. In general, the match directive will advance the FSM to a new state, and may in addition produce one or more output phonemes.

The formats for the match byte and directive byte are shown below.

Match byte:	+--+--+--+--+--+--+	
	0 match char	match character
	+--+--+--+--+--+--+	
	+--+--+--+--+--+--+	
	1 xxxx	NIL
	+--+--+--+--+--+--+	
Directive byte:	^+--+--+--+--+--+--+	
	0 0 phoneme	SC01 phoneme code
	+--+--+--+--+--+--+	
	+--+--+--+--+--+--+	
	0 1 0 n	n phonemes follow
	+--+--+--+--+--+--+	
	+--+--+--+--+--+--+	
	0 1 1 code	special action code
	+--+--+--+--+--+--+	(see Note 1, below)
	+--+--+--+--+--+--+	
	1 offset	offset to next state
	+--+--+--+--+--+--+	(see Note 2, below)

Note 1 -- Special actions include the following:

- delimiter (ignore).
- generate a marker.
- error (invalid token).

Note 2 -- The offset is used to update the translate table index as follows:

$\text{index} := (\text{index} + \text{offset}) \bmod 256$

This implies the following:

The maximum table size is 256 bytes.

All state transitions are in the forward direction, except for the transition to the top level state which is implicit in the specification of the FSM.

A state transition destination must be within 127 bytes of the source directive.

PART OF THE EFFORT OF
PROJECT SHOULD GO TO
DEVELOPING MACROS TO
FACILITATE TABLE CREATION

The FSM scanner operates with the input data as described below.

1. The translate table index is set to the beginning of the translate table (=0).

2. A new character is obtained.

3. The scanner scans the table linearly trying to find a match between the token character and one of the table entry match characters before a NIL entry is seen.

4. If a match is found, the scanner processes the match directive and does one of the following actions, based upon the type of directive.

- a. If the directive is a phoneme or multiple phonemes, the phoneme(s) are output and scanning proceeds at step 1.
- b. If the directive is a special action, the specified action is taken and scanning proceeds at step 1.
- c. If the directive is a table offset (new state), the table index is updated as specified and scanning proceeds at step 2.

5. If a match is not found (NIL found first), the scanner processes the match directive associated with the NIL and does one of the following actions, based upon the type of directive.

- a. If the directive is a phoneme or multiple phonemes, the phoneme(s) are output and scanning proceeds at step 6.
- b. If the directive is a special action, the specified action is taken and scanning proceeds at step 6.
- c. If the directive is a table offset (new state), the table

index is updated as specified and scanning proceeds at step 2.

6. The table pointer ~~is~~ is set to the beginning of the translate table (this is the same as step 1).

7. Scanning proceeds at step 3, using the character that produced the NIL match.

The translate table for the SC01 Symbolic Format is shown in Appendix D and the translate table for the World English Spelling Format is shown in Appendix E.

Appendix D -- Translate Table for SC01 Symbolic Form

LABEL (state)	MATCH CHAR	NEXT STATE	DIRECTIVE:	
			SPECIAL ACTION	PHONEME CODE
Start	'A'	Ax		
	'B'			B
	'C'	Cx		
	'D'	Dx		
	'E'	Ex		
	'F'			F
	'G'			G
	'H'			H
	'I'	Ix		
	'J'			J
	'K'			K
	'L'			L
	'M'			M
	'N'	Nx		
	'O'	Ox		
	'P'	Px		
	'R'			R
	'S'	Sx		
	'T'	Tx		
	'U'	Ux		
	'V'			V
	'W'			W
	'Y'	Yx		
	'Z'	Zx		
	' '		<delim>	
	'.'		<delim>	
	'.'		<delim>	
	'?'		<delim>	
	'-'		<delim>	
	'*'		<marker>	
	NIL		<error>	
Ax	'E'	AEx		
	'H'	AHx		
	'W'	AWx		
	'Y'			AY
	'1'			A1
	'2'			A2
Cx	NIL			A
	'H'			CH
Dx	NIL		<error>	
	'T'			DT
Ex	NIL			D
	'H'	EHx		
	'R'			ER
	'l'			E1
	NIL			E

Ix	'U'		IU
	'1'		I1
	'2'		I2
	'3'		I3
	NIL		I
Nx	'G'		NG
	NIL		N
Ox	'O'	OOx	
	'1'		O1
	'2'		O2
	NIL		O
Px	'A'	PAX	
	NIL		P
Sx	'H'		SH
	'T'	STx	
	NIL		S
Tx	'H'	THx	
	NIL		T
Ux	'H'	UHx	
	'1'		U1
	NIL		U
Yx	'1'		Y1
	NIL		Y
Zx	'H'		ZH
	NIL		Z
AEx	'1'		AE1
	'2'		AE2
	NIL		AE
AHx	'1'		AH1
	'2'		AH2
	NIL		AH
AWx	'1'		AW1
	'2'		AW2
	NIL		AW
EHx	'1'		EH1
	'2'		EH2
	'3'		EH3
	NIL		EH
OOx	'1'		OO1
	NIL		OO
PAX	'0'		PA0
	'1'		PA1
	NIL		
STx	'O'	STOx	<error>
	NIL		<error>
THx	'V'		THV
	NIL		TH
UHx	'1'		UH1
	'2'		UH2
	'3'		UH3
	NIL		UH
STOx	'P'		STOP
	NIL		<error>

Appendix E -- Translate Table for World English Spelling Form

The WES phoneme codes marked with an asterisk have yet to be mapped to their SC01 equivalents; some of these will be single SC01 codes and some will require multiple SC01 codes.

LABEL (state)	MATCH CHAR	NEXT STATE	DIRECTIVE:	
			SPECIAL ACTION	PHONEME CODE
Start	'A'	Ax		
	'B'			B
	'C'	Cx		
	'D'			D
	'E'	Dx		
	'F'			F
	'G'			G
	'H'			H
	'I'	Ix		
	'J'			J
	'K'			K
	'L'			L
	'M'			M
	'N'	Nx		
	'O'	Ox		
	'P'			P
	'R'			R
	'S'	Sx		
	'T'	Tx		
	'U'	Ux		
	'V'			V
	'W'	Wx		
	'Y'			Yl
	'Z'	Zx		
	' '			PA0
	'.'			PA0
	'.'			PA1
	'?'			PA1
	'-'		<delim>	
	'*'		<marker>	
	NIL		<error>	
Ax	'A'			AH1
	'E'			ae *
	'R'			ar *
	'U'			au *
Cx	NIL			a *
	'H'			ch *
Ex	NIL		<error>	
	'E'			ee *
	'R'			ER
	NIL			e *

Ix	'E'		ie *
	NIL		i *
Nx	'G'		NG
	'K'		nk *
	NIL		N
Ox	'E'		oe *
	'I'		oi *
	'O'		oo *
	'R'		or *
	'U'		ou *
	NIL		o *
Sx	'H'		SH
	NIL		S
Tx	'H'	THx	
	NIL		T
Ux	'E'		ue *
	'R'		ER
	'U'		uu *
	NIL		u *
Wx	'H'		wh *
	NIL		W
Zx	'H'		ZH
	NIL		Z
THx	'H'		THV
	NIL		TH

To: Distribution

Date: March 31, 1983

From: Harry B. Stewart

Subject: Speech Handler Interface

Here is a preliminary specification for a parallel bus speech handler to support the SC01 speech chip in future products. Comments are invited.

cc: Ken Balthaser
Sherwin Gooch
Rashid Khan
Joe Miller
Rich Pasco
Scott Scheiman



Introduction

There are several levels of speech representation which might be used to drive an SC01 speech device, four of which are shown below:

'hello' -- Text representation of "hello".

'heloe' -- World English Spelling of "hello".

'H EH1 EH2 L O1 PA0' -- Votrax symbolic phoneme representation of "hello".

1B 02 01 18 35 03 -- Votrax numeric phoneme representation of "hello", in hexadecimal.

It is my recommendation that we support the lower three levels shown, mapped exactly to the SC01 speech chip, the device used in the computer products currently being developed.

In order to allow for upward mobility of application and user software to other speech chips which might be utilized in Atari's future products, all speech data will be identified as being of a specific (device dependent) type and form. Future products will be able to accept data of their native type as well as translate data of older types to the closest equivalent sound in their new hardware.

Thus, for example, the three forms supported for the SC01 might be called '1P' (type #1, phonetic form), '1S' (type #1, symbolic form) and '1N' (type #1, numeric form), and future type/forms for the SC02 (or any other device) might be called '2P', '2S' and '2N'. Perhaps even a '2T' (type #2, text form) might be supported.

Hardware assumptions

The list items represent assumptions about the hardware interface, NOT desires.

SC01 speech chip.

One phoneme (6 bits) of external buffering, with separate direct control of the SC01 STB line.

Status bit and disableable IRQ interrupt associated with the SC01 A/R line.

No pitch/inflection control.

No volume control.

Handler functionality

The handler shall be able to output speech data (phonemes) when presented with data in either of two user formats: symbolic or numeric. In the symbolic format, a phoneme is represented by one to three ATASCII characters followed by a delimiter character; the handler converts each symbolic phoneme to a ~~single~~ ^{one or more} six bit numeric phoneme codes. In the numeric format, a phoneme is comprised of the lower six bits of an eight bit byte, which is assumed to contain the numeric phoneme code.

In addition, the handler shall have three output modes: direct, semi-buffered and fully-buffered. The output characteristics of the three modes are described below.

Direct -- output phoneme, wait for done, then return to caller.

Semi-buffered -- wait for prior phoneme done, output new phoneme, then return to caller.

Fully-buffered -- return to caller, then output all phonemes at interrupt level.

The caller may determine the completion of a phoneme string or synchronize himself to specific phonemes by any of several techniques, as listed below.

Phoneme counter in database -- The handler shall maintain a one byte counter in the PBI database area which shall be incremented as each phoneme is strobed into the SC01. This variable may be examined and/or altered by the caller at will.

FIFO control in database -- There shall be a FIFO associated with the handler which shall be used when phoneme output is to be fully buffered. This FIFO and its control elements are resident in the PBI database area and may be examined (but not altered) by the caller at will.

Markers -- There shall be one or more codes reserved for "markers". When a marker is about to be processed as phoneme data, the handler shall increment a marker variable in the PBI database area and produce a null phoneme (zero duration).

Which technique to use is a function of: 1) the output mode selected, 2) how well the caller records map to the synchronization points, and 3) whether the handler is called directly by the user or is invoked through CIO.

Handler/CIO interface

The device name is 'V:'

OPEN

The IOCB buffer pointer shall point to a sequence of ATASCII characters of the form shown.

V:<type><form><mode><EOL>

where: <type> = 'l'
 <form> = 'P' for World English Spelling phonetic form,
 'S' for Votrax symbolic form,
 'N' for Votrax numeric form,
 'U' for user specified symbolic form.
 <mode> = 'D' for direct mode,
 'S' for semi-buffered mode,
 'F' for fully buffered mode.

In response to an OPEN command, the handler initializes the output FIFO, resets all handler database variables and flags, and sends a STOP phoneme to the SC01. If the SC01 does not respond with an A/R status within xx milliseconds, an error status is returned.

If not specified, the type, form and mode default as shown below.

<type> = 'l'.
 <form> = 'N'.
 <mode> = 'D'.

Error conditions:

xx -- Votrax not responding.
 xx -- Invalid type (not equal to 'l').
 xx -- Invalid form (not equal to 'P', 'N', 'S' or 'U').
 xx -- Invalid mode (not equal to 'D', 'S' or 'F').

CLOSE

In response to a CLOSE command, the handler sends a STOP phoneme to the SC01.

Error conditions:

none

PUT BYTE

The handler accepts a single byte (character) of *phonetic form,* symbolic form

or numeric form data. The data is in the A register; the handler returns status in the Y register. The table below shows the processing that occurs for each byte, depending upon the form and mode.

FORM	MODE	PROCESS
N	D	The phoneme is output to the SC01, and the handler waits for the next SC01 request before returning.
	S	The handler waits for the SC01 to request a byte, outputs the phoneme to the SC01 and then returns.
	F	The byte goes to the output FIFO. If the FIFO is full, the handler loops waiting for an empty slot.
P,S,U	D	A token is assembled, translated to the proper numeric code, and then proceeds as in N-D above.
	S	a token is assembled, translated to the proper numeric code, and then proceeds as in N-S above.
	F	a token is assembled, translated to the proper numeric code, and then that byte is inserted to the output FIFO.

When operating in the fully buffered mode, the handler enables the SC01 interrupt, so that the handler's interrupt service routine can empty the FIFO.

Error conditions:

xx -- Unrecognizable text token.

SPECIAL

Change OPEN parms (need type, form and mode).

Define new FIFO (need location and size).

Define new translate table (need location).

include table format

INIT

TBD.

Speech data formats

World English Spelling -- "hello" is shown encoded in WES format, which requires 6 characters.

```
+--+--+--+--+
|H E L O E |
+--+--+--+--+
```

For the P form, the following characters are treated as word delimiters and produce pauses:

```
' ' -- space
',' -- comma.
'.' -- period.
'?' -- question mark.
```

For the P form, the hyphen is a token delimiter which allows the handler to unambiguously process the multi-character tokens. For example the word 'mishap' would be spelled 'mis-hap'.

For the P form, the ATASCII EOL is ignored.

The asterisk character is treated as a marker by the handler.

Symbolic -- "hello" is shown encoded in Votrax symbolic format, which requires 18 characters.

```
+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+
|H  E H 1  E H 2  L  O 1  P A 0|
+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+
```

For the S form, the following characters are treated as token delimiters and do not produce or alter phonemes:

```
' ' -- space
',' -- comma.
'.' -- period.
'?' -- question mark.
'-' -- hyphen.
<EOL> -- ATASCII end of line.
```

The asterisk character is treated as a marker by the handler.

Numeric -- "hello" is shown encoded in Votrax numeric format, which requires 6 bytes.

```
+--+--+--+--+--+--+--+
|1B 02 01 18 35 03|
+--+--+--+--+--+--+--+
```

All bytes received by the handler are truncated to 6 bit phoneme values and sent to the SC01, with the following two exceptions:

\$9B is an EOL and is ignored by the handler (null operation).
\$7F is the code for a marker.

Implementation details

Translate table pointer (user alterable).

Output FIFO buffer control:

Buffer empty.
Buffer full.
Buffer init.
Buffer test (empty, almost empty, almost full, full).

FIFO fills while in mid-record, handler hangs until room in FIFO.

RAM memory utilization (32 bytes available)

Phoneme counter [1].
Marker flag [1]
FIFO input index [1].
FIFO output index [1].
FIFO size [1].
FIFO pointer [2].
Default FIFO buffer [16].
Current data type [1].
Current data form [1].
Current output mode [1].
User translate table pointer [2].
Temporaries [4].

APPENDIX A -- World English Spelling Form

speech token	sound	Votrax equivalent
a	fAt	AE /2E or AE1/2F
aa	fAther	AH1/15
ae	pAY	A2 /05, A1 /06, A /20 or AY /21
ar	fAR	??
au	tAUt	AW1/13, AW2/30 or AW /3D
b	But	B /0E
ch	CHum	T /2A + CH /10
d	Dig	D /1E
e	sEt	EH3/00, EH2/01, EH1/02 or EH /3B
ee	sEE	Y /29, or E /2C or El /3C
er	gathER	ER /3A
f	Fat	F /1D
g	Gum	G /1C
h	Hat	H /1B
i	In	I3 /09, I2 /0A, I1 /0B or I /27
ie	tIE	??
j	Jam	D /1E + J /1A
k	Kit	K /19
l	Let	L /18
m	Met	M /0C
n	Net	N /0D
ng	siNG	NG /14
nk	siNK	??
o	On	??
oe	tOE	O /26, O2 /34 or O1 /35
oi	OIl	??
oo	tOO	U /28, IU /36 or U1 /37
or	fOR	??
ou	OUt	??
p	Pet	P /25
r	Run	R /2B
s	Set	S /1F
sh	SHed	SH /11
t	Tin	DT /04 or T /2A
th	THis	THV/38
thh	THing	TH /39
u	Up	UH1/32 or UH /33
ue	hUE	??
ur	fUR	ER /3A
uu	boOk	OO1/16 or OO /17
v	Van	V /0F
w	Win	W /2D
wh	WHeN	??
y	Yes	Y1 /22
z	Zoo	Z /12
zh	viSion	ZH /07

Appendix B -- Votrax Symbolic Form

speech token	sound	Votrax numeric equivalent
EH3	jackEt	00
EH2	Enlist	01
EH1	hEAvy	02
PA0	<pause>	03
DT	buTTer	04
A2	mAde	05
A1	mAde	06
ZH	aZure	07
AH2	hOnest	08
I3	inhibIt	09
I2	Inhibit	0A
I1	inhIbit	0B
M	Mat	0C
N	suN	0D
B	Bag	0E
V	Van	0F
CH	CHip	10
SH	SHop	11
Z	Z00	12
AW1	lAWful	13
NG	thiNG	14
AH1	fAther	15
OO1	lOOking	16
OO	bOOk	17
L	Land	18
K	triCK	19
J	JuDGe	1A
H	Hello	1B
G	Get	1C
F	Fast	1D
D	paid	1E
S	paSS	1F
A	dAY	20
AY	dAY	21
Y1	Yard	22
UH3	missIOn	23
AH	mOp	24
P	Past	25
O	cOld	26
I	pIn	27
U	mOve	28
Y	anY	29
T	Tap	2A
R	Red	2B
E	mEEt	2C
W	Win	2D
AE	dAd	2E

AE1	After	2F
AW2	sAlty	30
UH2	About	31
UH1	Uncle	32
UH	cUp	33
O2	fOr	34
O1	abOArd	35
IU	yOU	36
U1	yOU	37
THV	ThE	38
TH	THin	39
ER	bIRd	3A
EH	gEt	3B
E1	bE	3C
AW	cAll	3D
PA1	<pause>	3E
STOP	<stop>	3F